N90-23050

MULTIDISCIPLINARY EXPERT-AIDED ANALYSIS AND DESIGN (MEAD)

Thomas C. Hummel
Flight Dynamics Lab / FIGCA
Wright R&D Center
Wright-Patterson AFB, OH 45433

James H. Taylor
Control Systems Lab / KWD-209A
GE Corporate R&D
PO Box 8, Schenectady, NY 12301

ABSTRACT

The MEAD Computer Program (MCP) is being developed under the Multidisciplinary Expert-Aided Analysis and Design (MEAD) Project as a CAD environment in which integrated flight, propulsion, and structural control systems can be designed and analyzed. The MCP has several embedded computer-aided control engineering (CACE) packages, a user interface (UI), a supervisor, a data-base manager (DBM), and an expert system (ES). These modules have widely different interfaces and are written in several programming languages, so integrating them into a single comprehensive environment represents a significant achievement.

The supervisor monitors and coordinates the operation of the CACE packages, the DBM, the ES, and the UI. The DBM tracks the control design process. Models created or installed by the MCP are tracked by date and version, and results are associated with the specific model version with which they were generated. In addition, every model and result may have notes stored in the data base for user-supplied on-line documentation. The ES is used to relieve the control engineer from tedious and cumbersome tasks in the iterative design process. The UI provides the capability for a novice as well as an expert to utilize the MCP easily and effectively. Using the menu-driven access mode, a first-time MCP user may readily use the CACE packages, the ES, and the DBM. The expert user, on the other hand, may use MCP macros and two command entry modes to take advantage of the flexibility and extensibility of the MEAD environment.

The MCP version 2 (MCP-2.0) is fully developed for flight control system design and analysis. Propulsion system modeling, analysis, and simulation is also supported; the same is true for structural models represented in state-space form. The ultimate goal is to cover the integration of flight, propulsion, and structural control engineering, including all discipline-specific functionality and interfaces. This paper will discuss the current MCP-2.0 components and functionality.

1. INTRODUCTION

1.1 Motivation/Goal

Future aircraft designs will place more emphasis on the integration of aircraft control subsystems such as flight, propulsion, and structures. To a great extent, the design and analysis of these subsystems require similar analytical methods and software tools, yet the exchange of data and information among such disciplines is inefficient and time consuming during the conceptual and preliminary design phases because of varying notations, reference systems, and conventions. To effect this exchange, a computerized development environment is needed that contains a set of tools capable of accomplishing control system design and analysis tasks required by each discipline. This environment should allow automatic transfer of data between disciplines, thus enabling systems design and analysis to be accomplished efficiently in the preliminary design cycle. Such a system would allow each discipline to develop subsystems in the same time frame, thereby making integration feasible and producing designs that reduce aircraft complexity and improve overall performance. In addition, this environment should support users with CAD experience ranging from novice to expert, and provide rigorous tracking of the data generated throughout the design cycle. To meet these needs, the US Air Force initiated the Multi-disciplinary Expert-Aided Analysis and Design Program to define and create a computer-aided engineering environment to facilitate the design and analysis of modern flight control systems.

1.2 Approach

The MCP represents the culmination of three major tasks. Task 1 researched and documented the design methodologies for individual disciplines and for integrated flight, propulsion, and structural control (IFPSC). The second task built on Task 1 to develop the MEAD software requirements, specifications, and architecture for a computer program that would support the design methodologies identified in Task 1. Task 3 involved the implemention and testing of the first MEAD computer program (MCP-1.0) based on a subset of the definition developed in Task 2. MCP-1.0 was completed in March 1989. The test and evaluation period is in progress and is expected to be completed in late 1989. A general-purpose nonlinear simulation package (SIMNON) is being incorporated in the MCP as part of the second phase of the program which also includes substantial refinements and extensions; these will comprise MCP-2.0 which will enter the alpha test stage in September 1989.

1.3 Definition of the MCP

The MCP is a computer-aided control engineering environment for modeling, simulation, design, and analysis of linear and nonlinear airframes, engines, and structural models in state-space form. The CACE packages currently integrated into the MCP include the MATRIX$_X$® package for linear analysis and design, GENESIS†, ALLFIT†, and AUTOSPEC†; the SIMNON® program for nonlinear simulation, equilibrium determination, and linearization is being added at the present time. The MCP utilizes a supervisor that acts as the package integrator: All communications between the CACE packages, expert system (ES), data-base manager (DBM), and the user interface (UI) are coordinated by the supervisor.

The MCP tracks, documents, and dates models created and revised by the user. Notes may also be tagged to models if the user wishes. Results are documented and associated with a specific class of a model; these too may be annotated using the MCP Note Facility. Conditions specified for each simulation, analysis, or design result are also associated with the corresponding result file (e.g., duration of simulation, type of input, etc.). All of these capabilities are handled by the DBM automatically.

In terms of CACE activity, the MCP supports many general nonlinear and linear systems operations (see Section 2.2). Some functionality specific to flight control is also implemented in

® MATRIX$_X$ is a registered trademark of Integrated Systems, Inc, Santa Clara CA 95054; SIMNON is a registered trademark of Lund University, Lund, Sweden.

† GENESIS, ALLFIT, and AUTOSPEC are flight-control-specific packages developed by Northrop Corp. Aircraft Division (NCAD), Hawthorne CA 90250; see Section 2.1 for more details.

the MCP. For example, once a nonlinear aircraft system model is developed, the MCP has the ability to linearize it and run tolerance checks against the military flying qualities standard MIL-F-8785C. The MCP utilizes ALLFIT, AUTOSPEC (flying qualities assessment) along with the ES to do this. In the event that the system is out of tolerance, the MCP will iterate the flight control gains to bring the flight control system into compliance with specifications.

The user may operate the MCP using one or several different modes. For the more inexperienced CACE package user, the 'menu-driven' mode makes it possible to quickly and easily execute the basic CACE package functionity, as illustrated in Section 2.3. The more experienced user may access the CACE packages via menu-driven mode, either of two 'command-line' modes (MEAD commands and 'package' commands, i.e., $\mathrm{MATRIX_X}$ commands at present), or by using the 'MEAD Macro Facility', which allows an arbitrary mixture of MEAD commands, CACE package commands, and DCL commands to be executed within a single macro. MEAD commands are commands recognized by the MCP supervisor and converted internally into package commands; in some cases the translation is quite direct (e.g., the MEAD command for finding system model eigenvalues), in other cases MEAD commands are converted into a large number of package commands (e.g., the MEAD command for generating an input signal for a linear system in $\mathrm{MATRIX_X}$). The command-line modes would be used by the more experienced user whenever a MEAD macro would not make sense (i.e., when one or two commands are to be issued).

1.4 Outline

The remaining sections of this paper will present and discuss the following subject areas:

2.0 MEAD Computer Program Version 2.0 (MCP-2.0)
2.1 The MCP Architecture
2.2 MCP Functionality
2.3 The MCP User Interface
2.4 The MCP Data-Base Manager
2.5 The MCP Supervisor
2.6 MCP Expert-Aiding
2.7 MCP Hardware Requirements
3.0 Future Directions
4.0 Summary and Conclusion

2. MEAD COMPUTER PROGRAM VERSION 2.0 (MCP-2.0)

2.1 Architecture

The MEAD project approach to creating the MCP was to take maximum advantage of existing software modules. The resulting architecture of the MCP-2.0 is portrayed in Fig. 1. MEAD integrates and serves as a front end to the CACE packages, ES, and DBM. The CACE packages available in the MCP-1.0 software include: $\mathrm{MATRIX_X}$, GENESIS, ALLFIT, and AUTOSPEC. $\mathrm{MATRIX_X}$ provides MEAD with the linear design and analysis capability. GENESIS was developed by NCAD and provides the simulation, trimming and linearization of nonlinear airframes. ALLFIT and AUTOSPEC are also products of NCAD; these packages provide equivalent low-order fits to high-order flight control systems and military specification verification (MIL-F-8785C) respectively. All of these CACE packages were written in Fortran.

The UI, supervisor, DBM, the Delphi© ES shell, and the MCP rule bases were developed by General Electric Corporate R & D. The UI is implemented in the Computer/Human Interface Development Environment (CHIDE), supplied by GE Corporate R & D; this software in turn uses the Relational Object System for Engineering (ROSE) software, an object-oriented DBM developed by the Rensselaer Polytechnic Institute Center for Interactive Computer Graphics. ROSE and CHIDE are written in C. The supervisor coordinates all communications between the other software modules and translates MEAD commands into the equivalent core package commands in its package interface routines. The supervisor is written primarily in Ada, although the package interface routines involve some FORTRAN programming. The MCP's DBM consists of the ROSE DBM and the DEC® Code Management System (DEC/CMS®) software for version control and efficient model storage. The ES shell Delphi and portions of the rule bases are coded in LISP.

2.2 Functionality

The MEAD Computer Program provides an environment that supports modeling, simulation, analysis, and design of linear and nonlinear airframes, engines, structures (in state-space form), and control systems. The MCP is fully developed for flight control engineering. This means that the MCP has specialized software (i.e., ALLFIT, AUTOSPEC, GENESIS, and several rule bases in the ES) for flight control engineering. However, the MCP does not have the equivalent specialized software for propulsion and structural systems analysis and design.

The computer-aided control engineering packages in the MCP provide the analysis, design, simulation, trimming, and linearization capabilities. These capabilities and the functionality of the Expert System and DBM can be accessed via the Aircraft Integrated Design Environment (AIDE) menu tree (see Fig. 2). The corresponding top-level functionality that supports all three disciplines is outlined in the following sections.

2.2.1 Actively or Passively Manipulating the DBM - The data base can be examined by selecting the menu items 'Browse Projects', 'Browse Models', and 'Browse Results'. Browsing the data base reveals much information, e.g., classes and versions of models and components respectively, model and component type, creation and last modification dates, existence of associated notes, etc.; the Browse Models Screen in Fig. 3 illustrates the presentation of such material. Relations among components used in multiple models in the data base are indicated (such components are stored in one model and used elsewhere via 'linking'); also, the associations between results and components that have been created by 'modelizing' the results (an airframe model linearization result may be installed in the data base as a model component; we call this process modelizing) are tracked and displayed in the Description and Browse Results forms respectively. Active operations on the data base include deleting, updating, purging, and configuring models. ('Configuring' in MEAD terminology means loading component definition files into a core package and connecting them according to the user's specification so the model is ready to use for simulation, analysis, or design.) The MCP has a Note Facility for storing note files for projects, models, components, and results that are installed or generated in the system. Notes can consist of any information relevent to the corresponding data element; they are automatically time-stamped. The note files created and managed by this facility are thus an important vehicle for rigorous on-line documentation of the user's analysis and design effort.

© Delphi is a copyright of the General Electric Company, 1985.

® DEC and DEC/CMS are registered trademarks of Digital Equipment Corporation, Maynard, MA.

2.2.2 Defining Conditions for Simulations - A user can set parameters in nonlinear models, initialize the states of linear models, or define inputs to either linear or nonlinear system models. These capabilities are available in forms under the 'Def Condition' menu item, as shown in Fig. 2, by clicking 'Set Parameters', 'Init Variables', and 'Define Inputs', respectively.

2.2.3 Running Simulations - Simulations for linear and nonlinear models are handled in the same form, portrayed in Fig. 4. In that form there is an option to choose the type of model to simulate (i.e., linear or nonlinear). Parameters such as simulation time step and duration can also be set in the Simulation Form.

2.2.4 Trimming and Linearizing Nonlinear Models - A nonlinear model can be trimmed about the longitudinal, lateral-directional, or both axes in the full 6-degrees-of-freedom (6-DOF) case. When an axis set is chosen, a corresponding Trim Set-up form is generated, as shown in Fig. 5 for the longitudinal case. Within this form the user has the options to 'Set States' (defines the altitude, Mach number, and load along the lift vector for the desired flight condition), 'Set Limits' (defines the minimum, initial guess, and maximum values for the trim controls), and 'Set Constraints' (defines the trim states to be nulled and the tolerances). In the Linearize menu the options available are 'User Defined' (this option gives the user the ability to choose from a predefined list of possible inputs and outputs), and 'MIL Spec HOS' (this option automatically creates a linear model suitable for using ALLFIT to match the requirements for high-order flight control system model fitting - see Section 2.2.5).

2.2.5 Performing Linear Model Transforms - The Linear Model Transforms options consist of state-space to transfer function form, state-space to discrete-time state-space form, controllable part, observable part, minimal form, balanced form, reduced order, combine (which creates a single component from a multiple-component model), MIL Spec (military specification) Fit to obtain low-order equivalent linear systems for flying qualities assessment, and expert MIL spec fit. In many of these transformations, linear models are created and can be configured and studied. For a full description of these functionalities refer to the MEAD User's Guide [6].

2.2.6 Linear Analysis and Design - Many of the classical linear analysis and design techniques available in MATRIX_X have been incorporated in the MCP; these are indicated in the menu items under 'Lin Analysis' and 'Linear Design' in Fig. 2. The Lin Analysis menu also includes 'Flying Q Check', providing direct access to AUTOSPEC to assess the flying qualities of an aircraft. Rule bases are also used in conjunction with AUTOSPEC [3] and ALLFIT [4] (see Section 2.6); these are accessed via 'Expert FQ Chk' under Lin Analysis and 'Expert MIL Fit' under Lin Mdl Xform. Finally, in the Linear Design menu the item 'Exp Lead-Lag' invokes a rule base to generate lead and lag compensation to achieve specifications for band-width, gain margin and steady-state error (see section 2.6).

2.3 User Interface

The MEAD system integrates an ES, a DBM, and a variety of CACE packages in a single environment. As a stand-alone system, each of these pieces of software contains its own user interface. One objective of the MCP UI is to provide direct access to the various package functionalities, while relieving the user from needing to be intimately knowledgeable about the software packages as stand-alone systems and adapting to their various styles and syntaxes. This means, for example, that a person wishing to obtain a standard time history of a linear system should not need to know MATRIX_X commands for simulation and plotting; this is true using the MCP 'point-and-click' UI mode. However, the MCP UI should not unnecessarily confine the experienced user; to meet this requirement, the MCP 'Package Mode' provides the capability of executing $MATRIX_X$ commands within the MCP environment to perform any simulation and plotting activity allowed by that package, thus fully supporting the expert CACE package user. These examples illustrate how the UI objectives have been met by providing a multi-modal interface [7] which is overviewed below.

The MCP functionality can be accessed via point-and-click mouse operations on menu- and form-driven screens, as shown in Figs. 4 and 5. The top-level MEAD menu consists of Aircraft Integrated Design Environment (AIDE), Active, Command, Package, DCL, Macro, Help, and Exit (see also Fig. 2). The AIDE option was designed to provide the most conveniently accessible and functionally robust CACE capability. The 'Active' option displays the identifiers of the model(s) currently configured in the MCP Core Packages. (The user may have one linear model configured in $MATRIX_X$ and one nonlinear model in either SIMNON or GENESIS at any time in the session.) The 'Command' mode permits MEAD commands to be entered directly and dispatched to the supervisor, thus bypassing the UI. The 'Package' mode (currently operational for $MATRIX_X$ only) permits CACE package commands to be executed while operating under the MCP. VAX/VMS® DCL commands can be entered and passed directly to the operating system under the 'DCL' mode. The 'Macro' button accesses the MCP Macro Facility, which includes macro-execute mode and macro-edit mode and can support MEAD commands, package commands, DCL commands, or any combinations of these. The 'Help' facility is also menu-driven and has an extensive data base which can be accessed by subject menus. AIDE contains the linear design and analysis functions, simulation capability, nonlinear trimming and linearization routines, and DBM access, as outlined above.

As an example of the use of the MCP to obtain a standard result, the action flow for determining the controllability of a model is illustrated in Fig. 6. The 'AIDE' option is first chosen, which causes its first-level sub-menu to be displayed. The user then clicks 'Lin Mdl Xform', which brings up the second-level menu that includes the item 'Cntrl Part'. At this level, the form is created for defining and executing this operation. The entire menu tree down to the desired functionality becomes visible upon the selection of submenus. The user is given the option to execute the function (which transforms the configured continuous- or discrete-time model to obtain its controllable part), or the user can set the tolerance and/or perform the Grammian test. Once the controllable part has been determined the user has the option to 'Modelize' the result. This means the result will be installed in the data base as a model which may later be used for analysis and design. The user also can display and save the results at this point.

The UI provides an open, customizable, and flexible environment. The adept user of package commands, MEAD commands, and DCL can accomplish any task that is supported by the CACE packages. The user may also combine any of the these languages in a single MEAD macro to tailor the MCP to their preference. For example, a user can write macros to select projects in the DBM, configure models, and set up simulations using MEAD commands. It is also possible to take advantage of the $MATRIX_X$ command environment to achieve any result that can be calculated using that package alone; such results can still be stored and documented in the MEAD data base. This is just a small subset of the possibilities when using the MCP Macro Facility functionality.

® VAX and VMS are registered trademarks of Digital Equipment Corporation, Maynard, MA.

2.4 Data-Base Manager (DBM)

Data-base management requirements for CACE were determined as part of Task 1. Data-base elements were catalogued and categorized, and the relations among them were established. The MEAD data base is organized hierarchically with several levels of objects. The top-level, most general category of objects is Projects; below each project are found Models; and finally below each model are its Attributes (Descriptions and Results). Descriptions are comprised of representations of the components and their connection, type, etc.; the set of Results encompasses all data elements produced with the model (time-histories, eigenvalues, frequency response data, LQR designs, etc.). The user accesses a data base by selecting a project and operating on the corresponding unique data elements (e.g., configuring a model, displaying a result). The only "sharing" that can occur in the data base is at the component level: Models can share components with other models (via a mechanism called 'linking'); this allows a component to be maintained in one place in the data base, thus eliminating the trouble and risks involved in keeping and updating several copies of the same element. This scheme is illustrated in Fig. 5. Further details regarding the MEAD DBM are provided in [1, 2].

A user can think of this paradigm much like the DEC/VAX directory system for file organization. For example, a user might have set up subdirectories *[user.project1], [user.project2]* to manage the projects in Fig. 5, then *[user.project1.engine], [user.project1.airframe]*, etc. for the models, and perhaps even *[user.project1.engine.descript]* and *[user.project1.engine.results]* for the lowest level of the data base. In the case of MEAD, there is no need for the user to create subdirectories or track the random collection of files that accumulate therein.

While the CACE data-base categories outlined above are few in number and simple, there are several factors that complicate the DBM problem: models tend to change over the lifetime of the project, some results are also models (e.g., linearizations of nonlinear models or transformed linear models), and components tend to be used in several models yet they should be stored in one location to simplify their maintenance. The MCP DBM includes mechanisms to handle all of these situations with little or no burden to the user. This was in accordance with the specific design goal of providing DBM support with minimal changes in the way the IFPSC engineer works and with minimal added overhead. More specifically,

- *Versions and Classes* - The primary need for 'version control' in the conventional software engineering sense exists in the model level of the hierarchy. The DBM must be able to keep track of system models that evolve over time (e.g., as better modeling information becomes available or as preliminary modeling errors are corrected) so that each analysis or design result can be associated with the correct model instance. This observation motivated the use of a tool that tracks each *version* of a model component (e.g., airframe model) so that version = 1, 2, 3, ... refers to the orginal and subsequent refinements of this component model, and each *class* of a model (e.g., flight-control system) that incorporated the component.

- *Linking* - The CACE DBM requirements for tracking models also give rise to the need for non-redundant model management, since maintaining the integrity of the model level of the data base is nearly impossible if several copies of various components are separately stored and maintained. The MCP DBM supports this via *links*, which allows the engineer to maintain each component in one model (the 'home' model) and use it elsewhere by bringing it out of the home data base (DB) and incorporating it in other models.

- *Modelizing*- One remaining relation that complicates the hierarchical DB organization is that which associates a linearization as a result obtained using a nonlinear model with a linearization used as a model component. The same situation exists with regard to linear model transformation - for example, one may find the controllable part of a linear model, and desire to save this as both a result and model for further study by clicking 'Save' and 'Model' in Fig. 6. These associations are tracked in the MCP DBM using a mechanism called the *reference*. The engineer may inspect a linearization result and check the reference to see if it exists as a component in any model. From the other perspective, a linear model component may be checked to determine if it was obtained as a result generated with a particular nonlinear model and trace that result back to determine how it was obtained (e.g., at what flight condition). The value of a linear model is greatly reduced if component traceability in this sense cannot be assured.

An important secondary data element not depicted in Fig. 5 is the *condition specification*. This element contains information regarding operations performed on a model before a result is obtained. These include actions such as changing a parameter from its nominal value, specifying an initial condition and/or input signal before performing a simulation, defining a frequency list before obtaining Bode plot data, etc. The condition specification also records numerical conditions, such as setting a tolerance for determining controllability or observability. Selecting this data is critical, since it is the combination of model instance and condition specification that determines the result and thereby allows the engineer to document or repeat the result. Condition specifications are stored in the MEAD data base and may be recovered for any result that has been saved.

2.5 Supervisor

The supervisor monitors and coordinates all message handling from the user to the CACE packages, the ES, and the DBM. The supervisor interprets the user commands and translates them into package commands. Upon completion of a task/function, a package response is returned to the supervisor from a CACE package, the DBM, or the ES. The supervisor then translates this package response into standard form and conveys this information to the UI for display to the user. The DBM calls are all handled automatically based on the user's activity (e.g., creating models and results, editing models, annotating the data base, deleting data elements).

2.6 Expert-Aiding

The expert-aiding capability operates under the "control engineer's assistant" paradigm [8]. This means that the expert system is activated only when the user requests that it be used to perform a specific task. Once a user invokes the ES, results will then be generated and presented. The user has the opportunity to accept or reject the result from the ES; if the user chooses to reject the result, it is possible to respecify constraints/specifications and let the ES execute its rule base again.

The main purpose of the ES is to provide aid in clear-cut but substantial iterative control design procedures. Task 1 of the MEAD program identified several areas where expert-aiding could be used. Each area was evaluated and compared with the others in respect to time savings and feasibility. Four areas were selected for implementation; the resulting expert system consists of four main rule bases for Expert Military-Specification Fitting ('Expert MIL-Fit' in Fig. 2), Expert

Eigenvalue ('Expert Eigen'), Expert Flying Quality Assessment ('Expert FQ Chk'), and Expert Lead-Lag Compensator Design ('Exp Lead-Lag'). 'Expert MIL-Fit' takes advantage of the advanced frequency-dependent weighting functionality of ALLFIT by iterating these dependent weights to improve the low-order system fit for the axis selected (i.e., longitudinal) of the flight control system design. 'Expert Eigen' invokes Eigen Analysis, scrutinizes the results, and comments on them (e.g., 'The minimum damping ratio is zeta = 0.5148' for a stable system with complex poles). (This functionality was programmed only for software integration, demonstration purposes, and proof of concept.) 'Expert FQ Chk' provides flying qualities assessment using AUTOSPEC combined with control system design iteration to bring the flight control system into compliance with specifications.

2.7 Hardware Requirements

The MEAD Computer Program may currently be hosted only on the DEC VAX 11-xxxx family of computer systems and certain DEC workstations under the VMS operating system. The MCP UI requires use of a Tektronix® 4107 terminal (or a higher version), or of a personal computer with a suitable terminal emulator. This hardware platform was selected to support the Air Force on its existing computer facility.

3. FUTURE DIRECTIONS

Many extensions and refinements are being considered for future versions of the MCP. Selection will be based on the comments and recommendations of beta-test MCP users and practical issues of cost and implementability. Areas of high priority include execution speed, portability, flexibility (e.g., UI based on high-resolution graphics and windowing), user-friendly handling of linear models, and adding capabilities to cover specialized computer-aided control engineering for propulsion and structural control. Conversion of the operating system to UNIX® and porting the MEAD Computer Program to a workstation environment is anticipated in 1990. Other applications for expert-aiding are also under investigation.

4. CONCLUSION/SUMMARY

MEAD represents a new and innovative approach to CAE support for the control design process. The MCP is a supportive and flexible environment, designed to meet the user's needs in an appropriate and effective fashion regardless of the user's level of expertise. The important novel features of the MCP are the integrated engineering data-base manager, expert-aiding, and a multi-modal user-friendly interface. MEAD eliminates the need for the user to mentally track the data elements and design process during system development or to do so via manual means.

The MEAD Computer Program version 1.0 (MCP-1.0) represented the culmination of the MEAD Project's Task 3 effort. MCP-2.0 has been defined and implemented under an extension of the original effort. The MCP software is currently being tested and evaluated on several large program applications, e.g., the Wright R & D Center/TXAD High Altitude Long Endurance (HALE) program. Users' comments and evaluations are being recorded, and extensions and modifications are being planned on the basis of this feedback. The most important areas of future

® Tektronix is a registered trademark of Tektronix, Inc., Beaverton, OR.

® UNIX is a registered trademark of AT&T Bell Laboratories, Holmdel, NJ.

enhancement have been outlined in Section 3.

The MCP-2.0 is most fully developed for flight control systems engineering. Certain generic aspects of propulsion and structural systems analysis and design are also supported, including modeling, analysis, and simulation based on system models represented in state-space form. As a long-term goal, the MEAD project is striving for the total integration of flight, propulsion, and structural control engineering.

ACKNOWLEDGEMENTS

A major part of the work described above was performed for the Wright Research and Development Center, United States Air Force, Wright-Patterson AFB, Ohio 45433-6523, under contract F33615-85-C3611 by GE Corporate Research and Development (GE CR&D) as prime contractor, with primary subcontractor support from Northrop Corporation Aircraft Division (NCAD), GE Aircraft Engine (GE AE), and Aule-Tek Inc. The contributions of the following individuals are most gratefully recognized: NCAD - Dinesh Joshi, Pierre Wong, Peter Shaw, Juri Kalviste; GE AE - John Polley, Shrider Adibhatla; Aule-Tek - David Kassover, James Trojan, Michael Charbonneau, Albert Antoniotti; and GE CR&D - Tim Johnson, Magnus Rimvall, Hunt Sutherland, and Walt Dixon. In addition, Prof. Dean K. Frederick of Rensselaer Polytechnic Institute provided invaluable consulting services over the course of this project. Finally, numerous personnel at the USAF WRDC Flight Dynamics Laboratory have greatly influenced the course of this effort: 1st Lt Mark Schiller and Capt Mike Dunbar, and Mssrs. David Bowser, Stan Lash, John Bowlus, Tom Gentry, and Philip McKeehen.

REFERENCES

[1] Taylor, J.H., Nieh, K.H., and Mroz, P.A., "A Data-Base Management Scheme for Computer-Aided Control Engineering", *Proc. American Control Conference,* Atlanta, GA, June 1988.

[2] Mroz, P.A., McKeehen, P.D., and Taylor, J.H., "An Interface for Computer-Aided Control Engineering Based on an Engineering Data-Base Manager", *Proc. American Control Conference,* Atlanta, GA, June 1988.

[3] Wong, J.P., "AUTOSPEC Flying Qualities of Piloted Airplanes", Northrop Corporation Aircraft Division Control Systems Research, Hawthorne, CA, January 1989.

[4] Wong, J.P., "ALLFIT Equivalent Low-Order Systems Program", Northrop Corporation Aircraft Division Control Systems Research, Hawthorne, CA, January 1989.

[5] Taylor, J.H., and McKeehen, P.D., "A Computer-Aided Control Engineering Environment for Multi-Disciplinary Expert-Aided Analysis and Design (MEAD)", *Proc. National Aerospace and Electronics Conference (NAECON),* Dayton, Ohio, May 1989.

[6] Taylor, J.H., "User's Guide for MEAD Computer Program Version 1.0 (MCP-1.0)", General Electric CR&D, Schenectady, NY, March 1989.

[7] Rimvall, C.M., Sutherland, H.A., and Taylor, J.H., "GE's MEAD User Interface - a Flexible Menu- and Forms-Driven Interface for Engineering Applications", accepted for *Proc. CACSD '89,* Tampa FL, December 1989.

[8] Taylor, J.H., "Expert-Aided Environments for CAE of Control Systems" (plenary lecture), *Proc. 4th IFAC Symposium on CAD in Control Systems '88,* Beijing, PR China, August 1988.

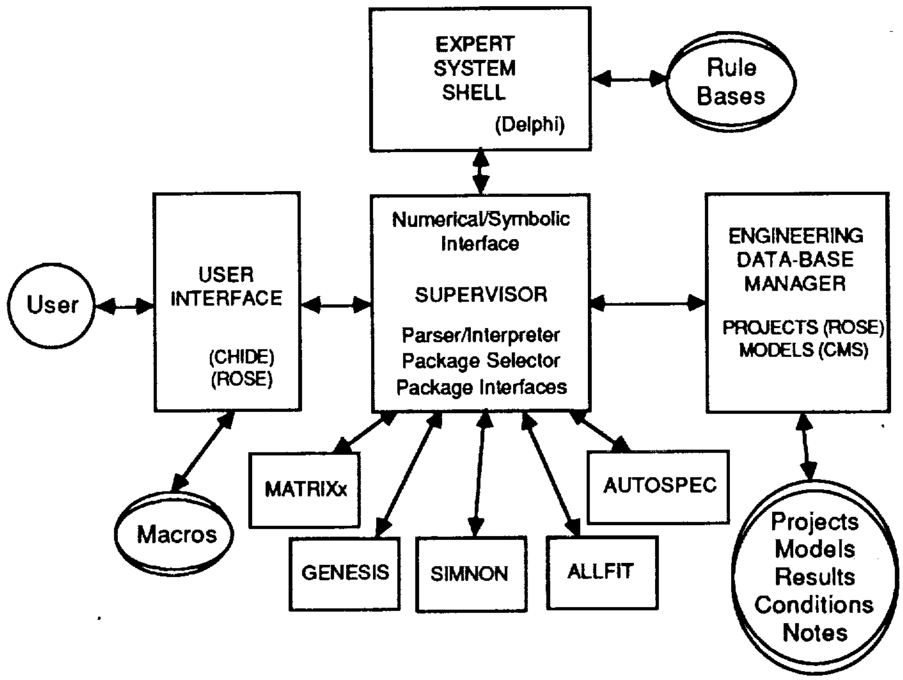

Figure 1. MCP Architecture

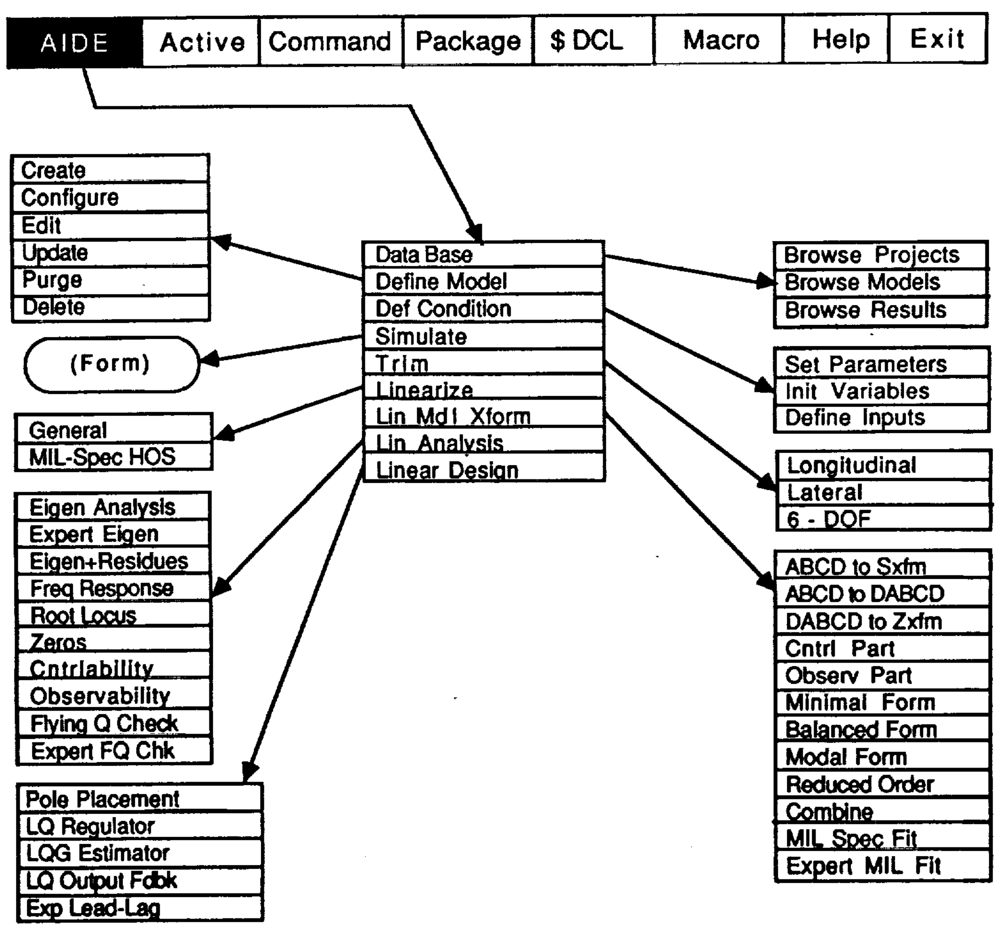

Figure 2. MCP Menu Tree

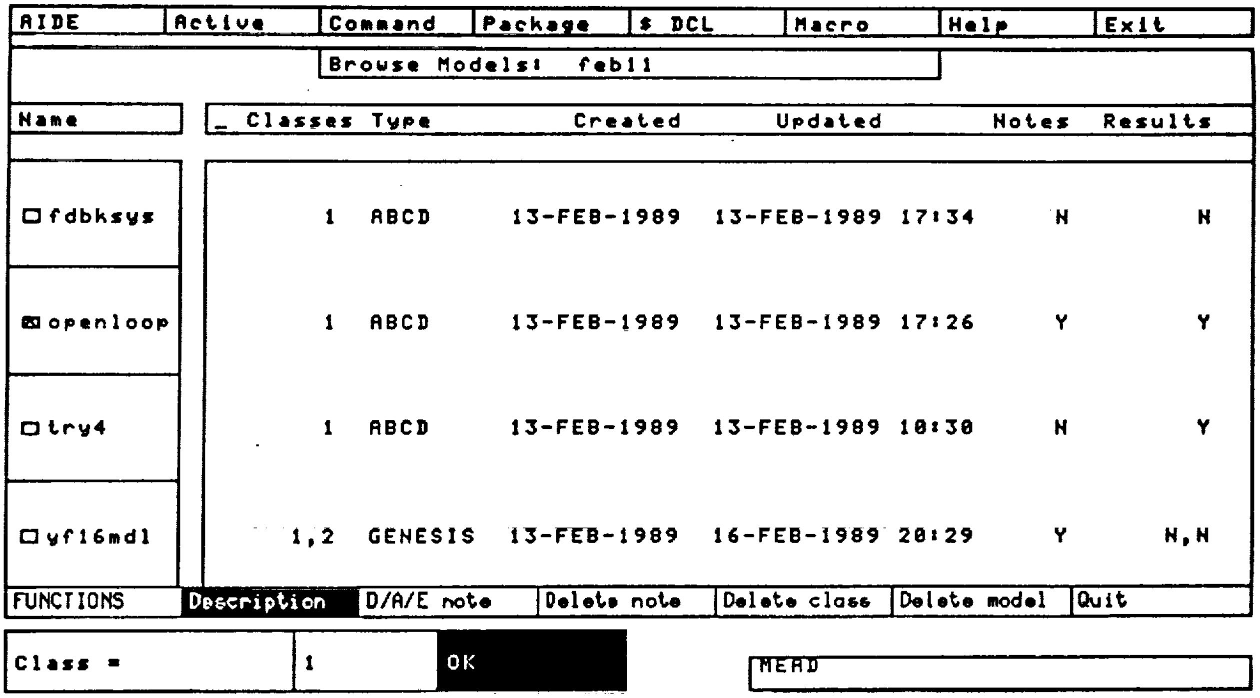

Figure 3. MCP Browse Models Screen

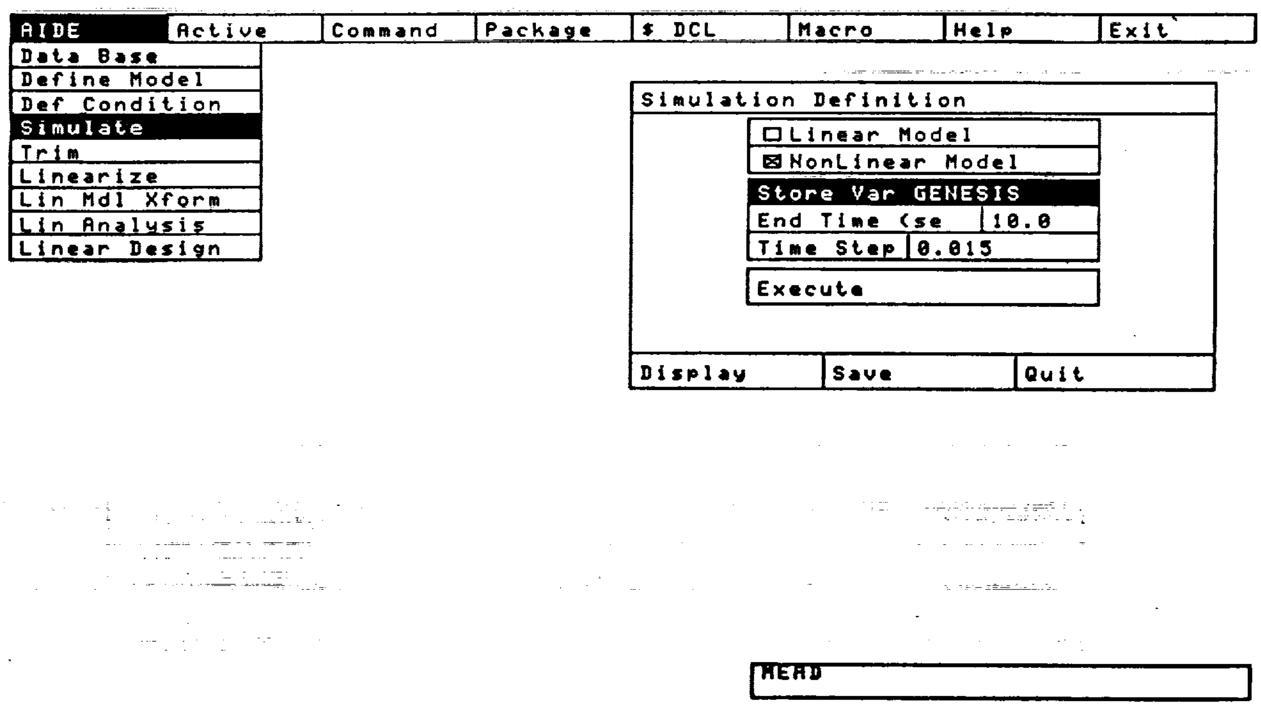

Figure 4. MCP Simulation Screen

ORIGINAL PAGE IS
OF POOR QUALITY

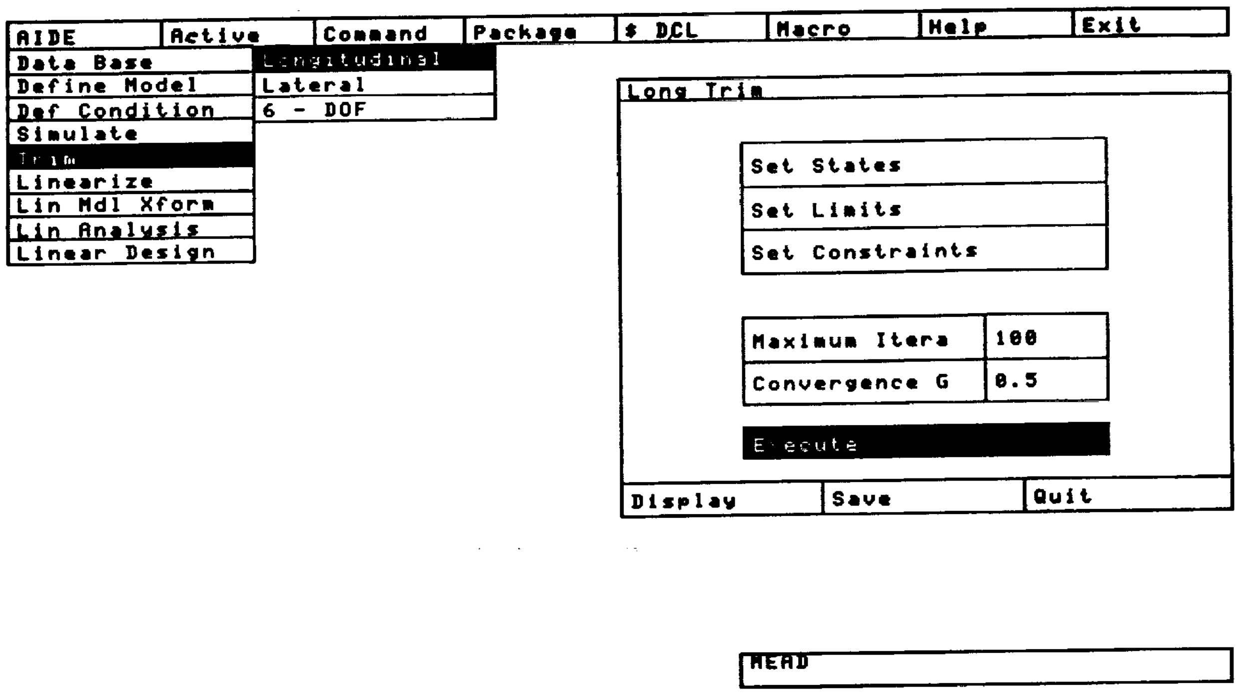

Figure 5. MCP Trim Set-up Screen

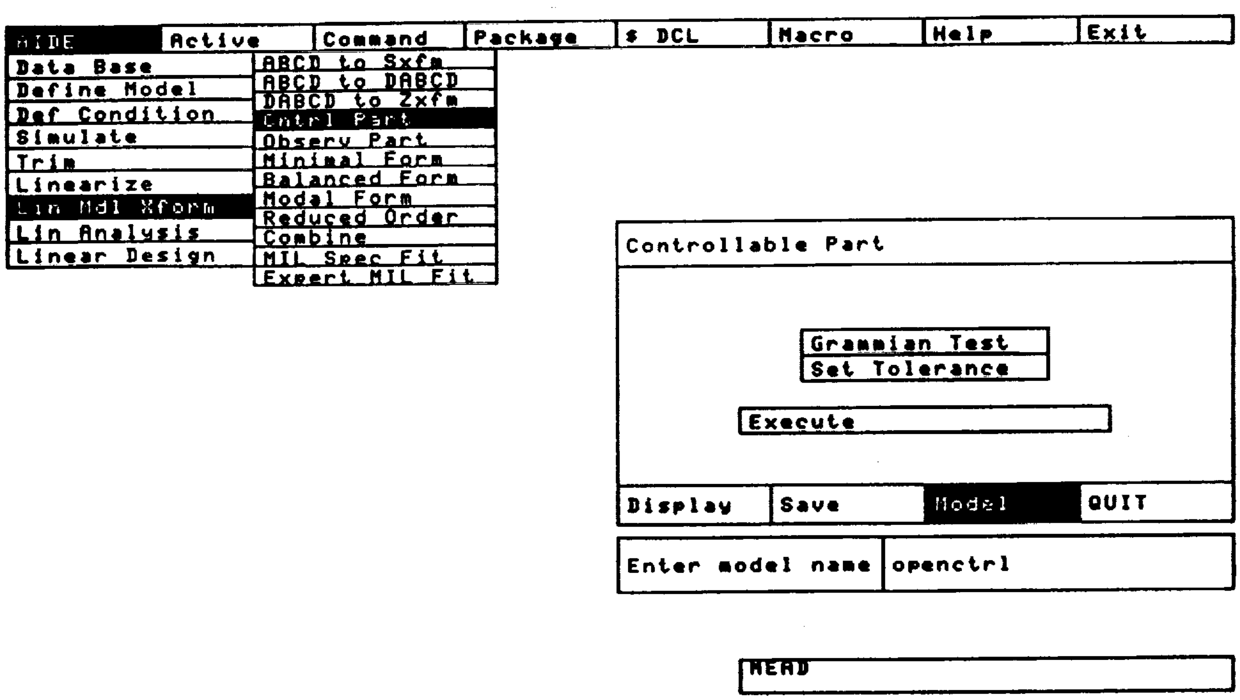

Figure 6. MCP Controllability Analysis Screen

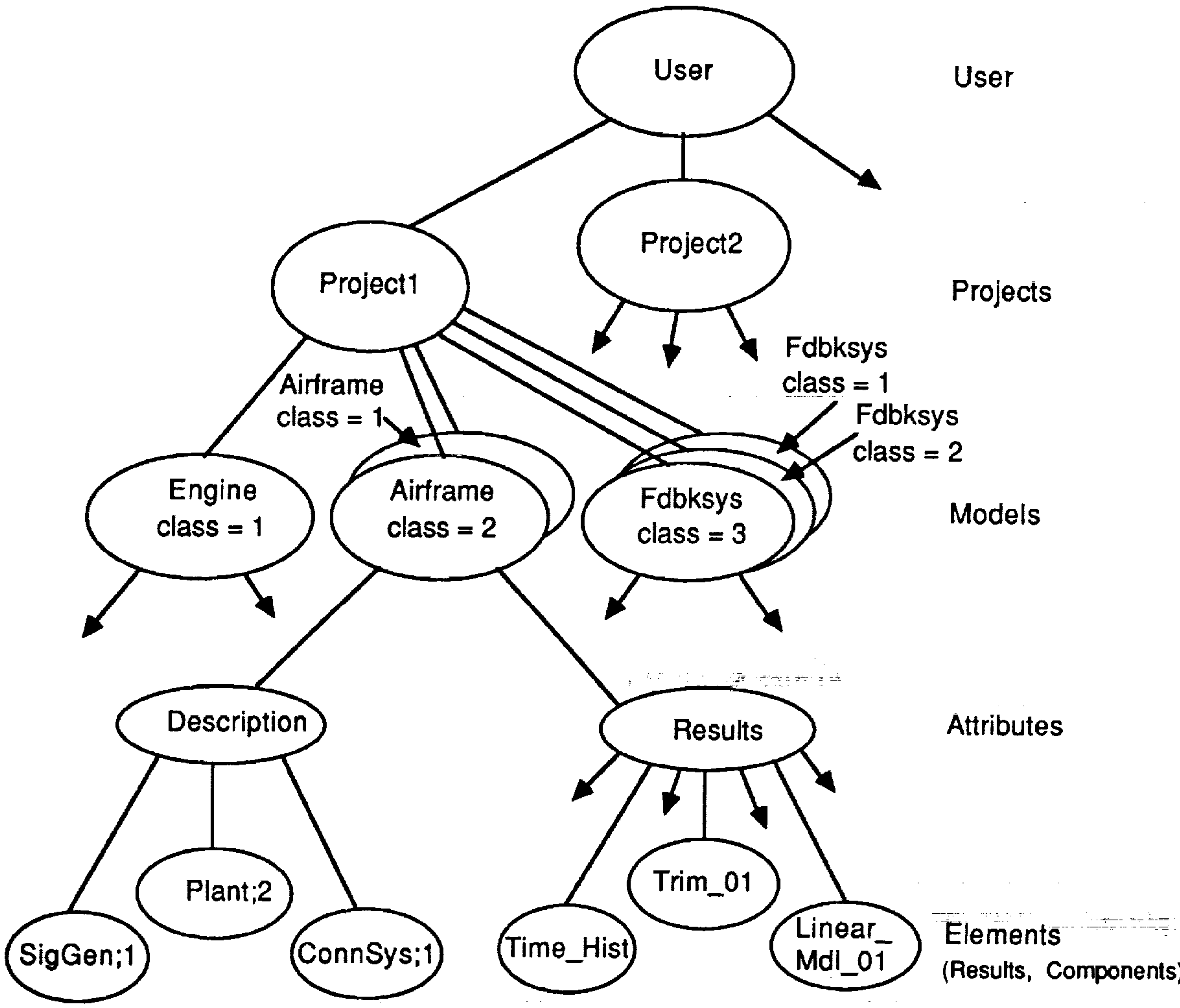

Figure 7. MCP Data-Base Version Control Schematic